Using linked data for semi-automatic guesstimation

Using Linked Data for Semi-Automatic Guesstimation

Jonathan A. Abourbih and **Alan Bundy** and **Fiona McNeill***
jabourbih@acm.org, bundy@staffmail.ed.ac.uk, f.j.mcneill@ed.ac.uk
University of Edinburgh, School of Informatics
10 Crichton Street, Edinburgh, EH8 9AB, United Kingdom

Abstract

GORT is a system that combines Linked Data from across several Semantic Web data sources to solve guesstimation problems, with user assistance. The system uses customised inference rules over the relationships in the OpenCyc ontology, combined with data from DBPedia, to reason and perform its calculations. The system is extensible with new Linked Data, as it becomes available, and is capable of answering a small range of guesstimation questions.

Introduction

The true power of the Semantic Web will come from combining information from heterogeneous data sources to form new knowledge. A system that is capable of deducing an answer to a query, even when necessary knowledge might be missing or incomplete, could have broad applications and appeal. In this paper, we describe a system that uses Linked Data to perform such compound query answering, as applied to the domain of *guesstimation*—the process of finding approximate (within an order of magnitude) numeric answers to queries, potentially combining unrelated knowledge in interesting ways (Weinstein and Adam 2008). The hypothesis that we are attempting to test is that *data contained in heterogeneous Linked Data sources can be reasoned over and combined, using a small set of rules, to calculate new information, with occasional human intervention.*

In this paper, we describe our implementation of a system, called GORT (Guesstimation with Ontologies and Reasoning Techniques), that does compound reasoning to answer a small range of guesstimation questions. The system's inference rules are implemented in Prolog using its Semantic Web module (Wielemaker, Schreiber, and Wielinga 2003) for RDF inference. The system uses the OpenCyc OWL ontology as a basis for its reasoning, and uses facts from other Linked Data sources in its calculations. This work is the result of a three-month-long Master's dissertation project (Abourbih 2009), supervised by Bundy and McNeill. Only the advent of interlinked data sources has made this system possible.

The rest of this paper is organised as follows: First, we describe prior work in the fields of deductive query answering and Semantic Web systems. Next, we outline the process of guesstimation. Then, we describe the organisation and implementation of GORT. Finally, we close with an evaluation of the system's performance and adaptability, and compare it to several other related systems. We also conclude with a brief section on future work.

*Funded by ONR project N000140910467

Literature Survey

Combining facts to answer a user query is a mature field. The DEDUCOM system (Slagle 1965) was one of the earliest systems to perform deductive query answering. DEDUCOM applies procedural knowledge to a set of facts in a knowledge base to answer user queries, and a user can also supplement the knowledge base with further facts. DEDUCOM also uses theorem proving techniques to show that the calculated answer follows logically from the axioms in the knowledge base. The QUARK/SNARK/Geo-Logica systems (Waldinger et al. 2003) exploit large, external data sources to do deductive query answering, however their focus was on yes/no and geographical queries and not numeric calculations. Cyc (Lenat 1995) is a large, curated knowledge base that can also apply deductive reasoning to derive new numeric facts in response to a user query. Wolfram|Alpha (`http://www.wolframalpha.com`) is a new system that combines facts in a curated knowledge base to arrive at new knowledge, and can produce a surprisingly large amount of information from a very simple query.

Query answering over the Semantic Web is a new field. The most developed system in this field is PowerAqua (Lopez, Motta, and Uren 2006), which uses a Semantic Web search engine to do fact retrieval, but it does not do any deduction based on that knowledge. There has been work on using Prolog's Semantic Web module in conjunction with custom-built ontologies to construct interactive query systems (Wielemaker 2009). Our GORT system, described in this paper, builds on prior work in knowledge-based systems and work on Semantic Web reasoning with Prolog. The system implements inference rules and plans for combining facts in Semantic Web data sources to derive new information in response to a user query.

Guesstimation

Guesstimation is a method of calculating an approximate answer to a query when some needed fact is partially or

completely missing. The types of problems that lend themselves well to guesstimation are those for which a precise answer cannot be known or easily found. Weinstein and Adam (2008) present several dozen problems that lend themselves to the method, and illustrate a range of guesstimation tactics. Two worked examples, of the type that GORT has been designed to solve, are given below.

Example 1. *How many golf balls would it take to circle the Earth at the equator?*

Solution. It is clear that this problem can be solved by calculating:

$$\frac{Circumference(Earth)}{Diameter(GolfBall)} \tag{1}$$

The solution requires the circumference of the Earth and the diameter of a golf ball. Weinstein and Adam (2008) show how the Earth's circumference can be deduced from common knowledge: a 5 hour flight from Los Angeles to New York crosses 3 time zones at a speed of about 1000 km/h, and there are 24 time zones around the Earth, yielding:

$$\frac{5\text{ h}}{3\text{ time zones}} \times 24\text{ time zones} \times 10^3\text{km/h}$$
$$\approx 4 \times 10^4\text{ km} \approx 4 \times 10^7\text{ m}.$$

Note that in (1), *Earth* is an individual object, but *GolfBall* does not denote any particular golf ball, but instead an arbitrary, or 'prototypical' golf ball. To distinguish between the class of objects called *GolfBall* and the arbitrary golf ball, we refer to the latter as $\epsilon GolfBall$. Continuing, we make an educated guess for $Diameter(\epsilon GolfBall)$ of about 4 cm. Substituting into (1) gives:

$$\frac{Circumference(Earth)}{Diameter(\epsilon GolfBall)} \approx\!\!> \frac{4 \times 10^7\text{m}}{4 \times 10^{-2}\text{m}} \approx\!\!> 10^9 \quad \square$$

In the solution to Example 1, the notation ϵS represents an arbitrary, 'prototypical' element from the set S. This notation and interpretation of our ϵ-operator is based on that of the Hilbert-ϵ operator (Avigad and Zach 2007).

Example 2. *What would be the total area occupied if all humans in the world were crammed into one place?*

Solution. We begin by summing up the area required by each human on Earth (2). If we assume that every human requires the same area, to within an order of magnitude, then this rewrites to:

$$\sum_{e \in Humans} Area(e) \tag{2}$$
$$\approx\!\!> Area(\epsilon Humans) \times \|Humans\|, \tag{3}$$

where $\epsilon Human$ is an arbitrary, representative element of the set of all *Humans*, and $\approx\!\!>$ represents a rewrite that is accurate to within an order of magnitude. The value for $Area(\epsilon Human)$ is a somewhat arbitrary choice (how close is *too* close?), so we make an educated guess of about 1 m^2. It is also generally-accepted common knowledge that $\|Humans\|$, the population of the Earth, is about 6×10^9. Substituting into (3) gives:

$$Area(\epsilon Humans) \times \|Humans\|$$
$$\approx\!\!> 1\text{ m}^2 \times (6 \times 10^9)$$
$$\approx\!\!> 6 \times 10^9\text{ m}^2 \quad \square$$

Guesstimation Plans

The examples from the previous section follow patterns that are common to a range of guesstimation queries. In this section, we generalise the examples above into plans that can be instantiated for other similar problems.

Count Plan

Example 1 asks for a count of how many golf balls fit inside the circumference of the Earth. This can be generalised to the form, "How many of some *Smaller* entity will fit inside of some *Bigger* entity?" and expressed as:

$$Count(Smaller,\ Bigger) \approx\!\!> \frac{F(Bigger)}{G(Smaller)}, \tag{4}$$

where $Count(Smaller,\ Bigger)$ represents the count of how many *Smaller* objects fit inside some *Bigger* object. Functions $F(x)$ and $G(x)$ represent measurements on an object, like volume, mass, length, or duration. The units of measurement must be the same. The variables *Smaller* and *Bigger* may be instantiated with a class, or an instance. In Example 1, this guesstimation plan was instantiated with the substitutions:

$$\langle Smaller/\epsilon GolfBall \rangle\text{, and}$$
$$\langle Bigger/Earth \rangle\text{.}$$

Size Plan

Example 2 asks for the total area that would be occupied by the set of all humans. This, too, can be generalised to: "What is the total of some property $F(e)$ for all instances e of a class S?" and expressed as:

$$\sum_{e \in S} F(e) \approx\!\!> F(\epsilon S) \times \|S\|, \tag{5}$$

where $\|S\|$ represents the count of elements in set S, and ϵS is an arbitrary, 'typical' element of S. In Example 2, this guesstimation plan was instantiated with the substitution $\langle S/Humans \rangle$.

Implementation

A system to perform compound query answering has a small set of requirements. First, the system needs some general background knowledge of the nature of the world. This knowledge is encoded in a *basic ontology* that forms the heart of the system's reasoning. Second, the system needs access to some set of numeric *ground facts* with which to compute answers. These facts may be available directly in the basic ontology, or from other linked data sources. Third, the system needs to have some *procedural knowledge*, and needs to know how to apply the procedures to advance a

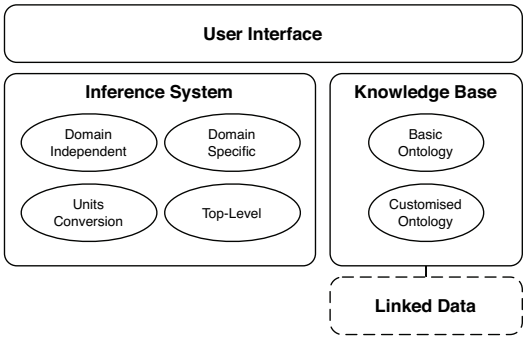

Figure 1: Module decomposition of the guesstimation system

solution. Fourth, in the course of solving a query, new information may need to be computed and recorded for later reuse. The system therefore requires some means of recording information. Finally, the system must make its facts and assumptions explicit, so that the user can determine the soundness of the solution. The design and implementation of GORT, shown in Figure 1, follows from these design requirements. The entire system is implemented in SWI-Prolog, using its Semantic Web and RDF modules (Wielemaker, Schreiber, and Wielinga 2003).

Basic Ontology and Ground Facts

The system possesses some general background knowledge about the relationships between a large variety of real-world concepts and entities. Currently, the system relies heavily on the relationships in the OpenCyc OWL ontology for this component. The system has access to ground facts in DBPedia (Auer et al. 2007), the GeoNames ontology, and the CIA Factbook ontology, via their links to OpenCyc. GORT relies on a locally-stored copy of these resources—there is no retrieval done at runtime from the Internet, although this is proposed as future work. Currently, the data sources have been retrieved by hand and are read in by the system at startup.

Inference System

The inference system consists of a set of Prolog predicates that implement the guesstimation plans described earlier, and some supporting inference rules for deriving intermediate calculation results.

Order of Magnitude Reasoning In line with typical Guesstimation techniques, GORT maintains the result of a calculation as an order of magnitude (OM) approximation, $m \times 10^x$, which is encoded as a Prolog predicate, $om(m, x)$. The system applies a set of standard rules for multiplying and dividing logarithms to perform calculations on OM representations (Nayak 1995).

Retrieval The system is capable of directly retrieving a fact from a Linked Data ontology where the URIs for the *Subject* and *Predicate* of the sought quantity are known. For example, the system can directly retrieve the population of Canada from the CIA Factbook ontology if it is provided with the URIs for *Canada* and *population*. The system is also capable of retrieving a fact for a given *Subject* and *Predicate* by following the graph of *owl:sameAs* relationships on *Subject* and *rdfs:subPropertyOf* relationships on *Predicate* until an appropriate *Object* is found. The Prolog Semantic Web module does not include logic for following *owl:sameAs* relationships; GORT implements a graph search algorithm to perform the search.

Arbitrary Objects In the example problems presented earlier, we found a need for an arbitrary (or prototypical) instance of a class in order to continue a calculation. When GORT is presented with a request in which the *Subject* is an instance of *rdfs:Class*, the system creates an instance of the class to represent an arbitrary element. For example, consider the golf ball of Example 1, where a request for $Diameter(GolfBall)$ might be expressed as:

$$\exists Value.\ rdf(ocyc\colon GolfBall,\ ocyc\colon diameterOfObject,\ Value).$$

GORT detects that *ocyc:GolfBall* is an instance of *rdfs:Class*, and asserts the following RDF triple, where $\_\_bnode1$ is a fresh blank node that corresponds to $\epsilon GolfBall$:

$$rdf(\_\_bnode1, rdf\colon type, ocyc\colon GolfBall)$$

GORT can then apply other tactics and methods to obtain an appropriate value for the new $\epsilon GolfBall$.

In the case where there already are other known instances of the class, the system can compute an appropriate value by finding the average over all instances of the class. For example, if the query requires $Height(\epsilon Human)$, GORT will compute the average *Height* for all known *Human*s in the linked data, and use that value for $Height(\epsilon Human)$.

Domain-Specific GORT also contains methods for calculating facts from domain-specific knowledge. When searching for a value for a geometric property, like volume or surface area of a solid, GORT applies the knowledge in OpenCyc to determine appropriate plans to apply given the available information. For example, if GORT requires $Circumference(Earth)$, but only has access to $Radius(Earth)$, it detects that *Earth* is a round object (and therefore has a circumference), and compute the appropriate value. GORT can also aggregate populations of countries to find the populations of continents, based on the information in OpenCyc and the CIA World Factbook. GORT also contains logic to convert between scale units (e.g. metres to kilometres).

User Interaction If no other set of tactics or methods have successfully found a result, GORT asks the user to make an educated guess.

Customised Ontology

The system constructs a new customised ontology for each user query. It acts as a repository for the system to store the calculation results of each step in a query solution. If the

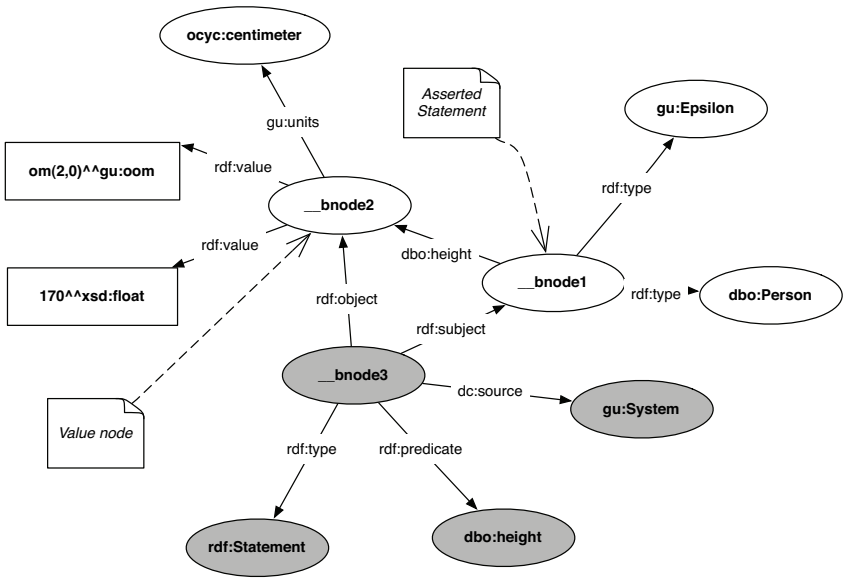

Figure 2: An example statement in the customised ontology, declaring that the height of a typical person is 170 cm. The reification of the statement and associated *dc:source*, shown in grey, explains that the source of the statement is *gu:System*, denoting a calculated result.

query requires the volume of a typical golf ball, for example, but the system only has access to a diameter, the system calculates the volume and stores the result in the customised ontology. Because the customised ontology records intermediate calculation results, it also makes explicit all of the computed knowledge and assumptions that the system has used to answer a query. Furthermore, the system reifies each statement, using the RDF reification mechanism, and records how each statement was derived. This reification mechanism opens the possibility of future work in which the system could estimate the reliability of items in its customised ontology based on their sources or underlying assumptions. An example entry in the customised ontology, giving the height of a typical human, $Height(\epsilon Human)$, is shown in Figure 2.

Evaluation

The system has been evaluated along two dimensions: its performance, and its adaptability to new content. The system's *performance* was measured by its ability to accurately answer a query to within an order of magnitude, to successfully produce an answer to a query, and its efficiency. The system's *adaptability* was measured by its ability to incorporate new linked data into its guesstimation process, when the data becomes available. These criteria are similar to the ones that have been used to evaluate other Semantic Web systems (Lopez et al. 2008).

A set of five queries was created to evaluate the system. Two of the queries were selected from Weinstein and Adam 2008; the remaining three were developed to be similar in style to the first two, but to cover different knowledge domains. Our analysis shows that the system produced accurate responses to within an order of magnitude for four of the five queries attempted. The failure case is analysed in detail below. The system successfully produced an answer in all cases where sufficient data existed in the knowledge base, or invoked the *user interaction* method of last resort when necessary information was missing and could not be derived. The system produced answers to most queries in 10–15 seconds, over a knowledge base of approximately 3 million RDF triples. The adaptability of the system was gauged by its ability to modify its execution plan when new RDF triples appeared in the knowledge base, simulating the appearance of new knowledge in the web of linked data. To measure this, we took a query for which there was insufficient knowledge to calculate a solution, and introduced an appropriate set of triples from DBPedia. Once the new knowledge had been introduced, the system was able to successfully compute a result.

The failed query asked, "What is the total volume of all human blood cells on Earth?" which is problem 4.2 from Weinstein and Adam (2008). The system produced an answer of $10^9\ \mathrm{m}^3$, but the expected result was $3 \times 10^7\ \mathrm{m}^3$. To determine the source of the error, the query plan was examined in detail. The query, as entered by the user, instructs the system to perform the following calculations, applying the *Count Plan* and *Size Plan*, as appropriate:

$$\sum_{e \in BloodCell} Volume(e)$$
$$\approx\!\!> Volume(\epsilon BloodCell) \times Count(BloodCell,\ Person) \times Count(Person,\ Earth)$$

$$Count(BloodCell,\ Person) \gtrapprox \frac{Volume(\epsilon Person)}{Volume(\epsilon BloodCell)} \quad (*)$$

$$\gtrapprox 10^{-15}\ \mathrm{m}^3 \times \frac{2 \times 10^5\ \mathrm{cm}^3}{10^{-15}\ \mathrm{m}^3} \times 6 \times 10^9 \quad \gtrapprox 10^9\ \mathrm{m}^3$$

The error occurs in the step indicated by (*), in which the system has instantiated the *Count Plan* and applied rewrite rule (4). One assumption of the *Count Plan* is that the larger object (*Person*) is entirely composed of the smaller object (*Blood Cells*). In this case, that assumption is clearly violated—no person is entirely composed of blood. However, the system lacks the knowledge to make that determination. It should include some adjustment factor to account for the other materials that compose a typical human body, like bones and tissue. A future extension to the system may be able to detect this situation and adjust its calculations appropriately.

Related Work

There are several Semantic Web knowledge-based systems that share some common characteristics with GORT, although there are none that take its approach to applying rules to derive new facts from linked data. In this section, we compare GORT with PowerAqua, a Semantic Web-based question answering system (Lopez, Motta, and Uren 2006); Cyc, a general-purpose common-sense knowledge base and reasoning system (Lenat 1995); and Wolfram Alpha, a system that calculates answers to numerical questions spanning a broad range of topics. Motta and Sabou (2006) propose a framework for evaluating next-generation Semantic Web applications, which we adopt here. The framework evaluates a system along four dimensions: the system's ability to reuse Semantic Web data; whether the system relies on a single monolithic ontology, or can integrate multiple ontologies; how well the system adapts to new Semantic Web resources at the user's request; and how the system scales to large data sets or high use. We have adopted their framework here for our comparison of the above systems with GORT.

Reuse of Semantic Web Data

A large-scale Semantic Web system must be able to reuse knowledge contained in sources outside of the system boundary. To determine whether a system meets this criterion, we examine how each of the systems can integrate new Semantic Web resources into its reasoning. The first system we examine is PowerAqua. As a Semantic Web-based question answering system, PowerAqua relies on data from distributed Semantic Web resources, accessed via a Semantic Web search engine. The system answers user queries based on data gathered from a large number of ontologies, which it does dynamically at runtime. Like PowerAqua, GORT also reuses data from multiple Semantic Web resources, taking advantage of the linked data relationships between them. GORT relies on locally-cached copies of its data sources, with on-line retrieval left as future work. Both Cyc and Wolfram Alpha, have limited ability to reuse Semantic Web data. Cyc contains a large curated knowledge base, and does not allow the dynamic reuse of information from Semantic Web sources, although techniques exist for mapping Cyc concepts to external concepts (Reed and Lenat 2002). Wolfram Alpha also uses a large, curated knowledge base. As a proprietary system, there is little information available on its functioning. That system does not explicitly make use of Semantic Web resources

Single- vs. Multiple-Ontology

Next, we consider whether each system relies only on a single ontology or is adaptable to multiple ontologies. A multi-ontology system uses ontologies from different linked data or other Semantic Web resources to perform its reasoning; a single-ontology system assumes that it operates only within the confines of its own closed domain. PowerAqua works with multiple ontologies at the same time, which it discovers with its Semantic Web search engine, and integrates them at runtime to answer user queries. Again, like PowerAqua, GORT does not make any assumptions about the origins of its knowledge, nor does GORT rely on a single, centralised ontology for its reasoning. Instead, GORT can incorporate knowledge from multiple linked data sources, providing that the connections between concepts in each ontology already exist. As proprietary systems with hand-curated knowledge bases, both Cyc and Wolfram Alpha are single-ontology.

Openness

Third, we consider each system's openness to newly-introduced Semantic Web resources. PowerAqua is capable of incorporating new ontologies into its query answering, at the user's request and without additional configuration. Further investigation is required to determine how open GORT is to new ontologies. The incorporation of generic plans into the system's inference engine should allow it to adapt to new resources as they become available. The system relies heavily on the OpenCyc OWL ontology, and thus any new linked data need to have existing mappings, either directly or through an intermediary ontology, into OpenCyc. Neither Cyc nor Wolfram Alpha can adapt readily to new external data sources—a consequence of their single-ontology approach. Although techniques exist for mapping new ontologies into Cyc (Masters 2002), they cannot be automatically retrieved and mapped at the user's request.

Scalability

Finally, we consider each system's ability to scale to large data sets. The Semantic Web currently contains billions of RDF triples, and Semantic Web systems of the future will derive their intelligence primarily from their ability to manage large volumes of data (d'Aquin et al. 2008). PowerAqua has been designed with the large-scale Semantic Web in mind, and its query performance has been tested against large data sets. Results from a recent evaluation show that PowerAqua can take up to 30 minutes to answer some questions (Lopez et al. 2008). PowerAqua has access to significantly more data and more sophisticated inference algorithms than does GORT. That notwithstanding,

the 10–15 second response time over 3 million triples suggests that GORT is scalable to several million more, based on prior evaluations of the Prolog Semantic Web module that the system employs (Wielemaker 2009). Cyc contains a large knowledge base, on the order of 1 million axioms and 100,000 concepts as of 1995 (Lenat 1995). Despite Cyc's size, it is small in comparison to the projected size of the Semantic Web. Cyc's adaptability to Semantic Web resources has previously been investigated, but no work has been done to evaluate its performance when large external data sets are incorporated into its knowledge base. For Wolfram Alpha, there is no public estimate of the amount of data that it accesses, but the range of questions that it can answer suggests a very large knowledge base.

Conclusions and Future Work

The Semantic Web, and the Linked Data initiative in particular, offer a large amount of knowledge that can be used to build intelligent systems. d'Aquin et al. (2008) point out that the most successful next-generation Semantic Web systems will be the ones that have the largest amount of data and are capable of working with it in interesting ways. GORT works with large linked data sets to deduce numeric answers to queries, by applying a small set of rules in combination. Thus, we set out to show that *data contained in heterogeneous linked data sources can be reasoned over and combined, using a small set of rules, to calculate new information, with occasional human intervention.* We tested this hypothesis by constructing a system called GORT that combines data from OpenCyc, DBPedia, and the CIA World Factbook to calculate new information in response to user queries. The system relies on ontological relationships in OpenCyc and the links between OpenCyc concepts and numeric facts in DBPedia and the CIA World Factbook. Based on our evaluation of the system, it has been shown that it is capable of responding accurately to a small range of queries.

Currently, the system relies on locally-stored extracts from linked data sources. We intend to extend GORT so that it retrieves knowledge directly from the Web by following Linked Data URIs, as needed. GORT's user interface is also very rudimentary. Currently, queries are posed by writing a short Prolog predicate using one of the top-level guesstimation plans. Any time GORT needs to ask the user a question, this is done through a command line interaction; to increase GORT's general appeal, a graphical user interface could be added. As the Semantic Web grows, conflicting facts are likely to exist; therefore, GORT needs a mechanism to determine which facts are more appropriate to use for a calculation, and which can be disregarded. Support for such a feature is provided by the reification of each statement in the current implementation, but the system does not make use of that knowledge at this time. GORT is under continuing development, currently as an undergraduate project at the University of Edinburgh.

References

Abourbih, J. A. 2009. Method and system for semi-automatic guesstimation. Master's thesis, University of Edinburgh, Edinburgh, Scotland.

Auer, S.; Bizer, C.; Kobilarov, G.; Lehmann, J.; Cyganiak, R.; and Ives, Z. 2007. DBpedia: a nucleus for a web of open data. In *The Semantic Web*, volume 4825/2007 of *LNCS*. Berlin: Springer. 722–735.

Avigad, J., and Zach, R. 2007. The epsilon calculus. Retrieved from `http://plato.stanford.edu/entries/epsilon-calculus/` on 4 July 2009.

d'Aquin, M.; Motta, E.; Sabou, M.; Angeletou, S.; Gridinoc, L.; Lopez, V.; and Guidi, D. 2008. Toward a new generation of semantic web applications. *IEEE Intelligent Systems* 23(3):20–28.

Lenat, D. B. 1995. Cyc: a large-scale investment in knowledge infrastructure. *Communications of the ACM* 38(11):33–38.

Lopez, V.; Guidi, D.; Motta, E.; Peroni, S.; d'Aquin, M.; and Gridinoc, L. 2008. Evaluation of Semantic Web Applications. OpenKnowledge Deliverable D8.5, Knowledge Media Institute, The Open University, Milton Keynes, England. Retrieved from `http://www.cisa.inf.ed.ac.uk/OK/Deliverables/D8.5.pdf` on 10 August 2009.

Lopez, V.; Motta, E.; and Uren, V. 2006. PowerAqua: fishing the semantic web. In *The Semantic Web: Research and Applications*. Berlin: Springer. 393–410.

Masters, J. 2002. Structured knowledge source integration and its applications to information fusion. In *Proceedings of the Fifth International Conference on Information Fusion*, volume 2, 1340–1346. Parsippany, NJ: IEEE.

Motta, E., and Sabou, M. 2006. Next generation semantic web applications. In *The Semantic Web – ASWC 2006*. Berlin: Springer. 24–29.

Nayak, P. P. 1995. Order of magnitude reasoning. In *Automated Modeling of Physical Systems*, volume 1003/1995 of *LNCS*. Berlin: Springer. 145–167.

Reed, S. L., and Lenat, D. B. 2002. Mapping ontologies into Cyc. In *Proc. AAAI Conference 2002 Workshop on Ontologies for the Semantic Web*.

Slagle, J. R. 1965. Experiments with a deductive question-answering program. *Communications of the ACM* 8(12):792–798.

Waldinger, R.; Appelt, D.; Fry, J.; Israel, D.; Jarvis, P.; Martin, D.; Riehemann, S.; Stickel, M.; Tyson, M.; Hobbs, J.; et al. 2003. *Deductive question answering from multiple resources*. Menlo Park, CA: AAAI Press.

Weinstein, L., and Adam, J. A. 2008. *Guesstimation: Solving the World's Problems on the Back of a Cocktail Napkin*. Princeton, NJ: Princeton University Press.

Wielemaker, J.; Schreiber, G.; and Wielinga, B. 2003. Prolog-Based infrastructure for RDF: scalability and performance. In *The Semantic Web - ISWC 2003*, volume 2870/2003 of *LNCS*. Berlin: Springer. 644–658.

Wielemaker, J. 2009. *Logic Programming for Knowledge-Intensive Interactive Applications*. Ph.D. Dissertation, Universiteit van Amsterdam.